A Novel Multi-objective Evolutionary Algorithm Based on Space Partitioning

Xiaofang Wu[1,2], Changhe Li[1,2,⋆], Sanyou Zeng[3], and Shengxiang Yang[4]

[1] School of Automation, China University of Geosciences, 430074 Wuhan, China
[2] Hubei Key Laboratory of Advanced Control and Intelligent Automation for Complex Systems, 430074 Wuhan, China
wuxiaofang@cug.edu.cn, lichanghe@cug.edu.cn
[3] School of Mechanical Engineering and Electronic Information, China University of Geosciences, 430074 Wuhan, China
sanyouzeng@gmail.com
[4] School of Computer Science and Informatics De Montfort University Leicester, LE1 9BH, United Kingdom
syang@dmu.ac.uk

Abstract. To design an effective multi-objective optimization evolutionary algorithms (MOEA), we need to address the following issues: 1) the sensitivity to the shape of true Pareto front (PF) on decomposition-based MOEAs; 2) the loss of diversity due to paying so much attention to the convergence on domination-based MOEAs; 3) the curse of dimensionality for many-objective optimization problems on grid-based MOEAs. This paper proposes an MOEA based on space partitioning (MOEA-SP) to address the above issues. In MOEA-SP, subspaces, partitioned by a k-dimensional tree (kd-tree), are sorted according to a bi-indicator criterion defined in this paper. Subspace-oriented and Max-Min selection methods are introduced to increase selection pressure and maintain diversity, respectively. Experimental studies show that MOEA-SP outperforms several compared algorithms on a set of benchmarks.

Keywords: Multi-objective optimization, kd-tree space partitioning, Max-Min method

1 Introduction

Multi-objective optimization problems (MOPs) widely exist in engineering practice. There is no single optimal solution, but a set of trade-off optimal solutions, because of the conflict between objectives. Multi-objective evolutionary algorithms (MOEAs), with the ability to obtain a set of approximately optimal solutions in a single run, have become a useful tool to solve MOPs.

Over the past decades, with the development of evolutionary multi-objective optimization (EMO) research, domination-based and decomposition-based algorithms have attracted many researchers.

⋆ The corresponding author.

One of the most classical domination-based MOEAs is the nondominated sorting genetic algorithm (NSGA-II) [1]. NSGA-II selects offspring according to elitist nondominated sorting and density-estimation metrics. However, it takes too many resources on the convergence during evolution, which results in loss of diversity.

One of the most classical decomposition-based MOEAs is the MOEA based on decomposition (MOEA/D) [2], which decomposes a MOP into multiple single-objective subproblems. Although it simplifies the problem, it still suffers several issues. Decomposition approaches are very sensitive to the shapes of Pareto front (PF). For example, Weighted Sum (WS) can not find Pareto optimal solutions in the nonconvex part of PF with complex shape. On extremely convex PF, a set of solutions obtained by Tchebycheff (TCH) are not uniformly distributed.

To address the sensitivity issue discussed above, some researchers have proposed the grid-based MOEAs, such as the grid-based evolutionary algorithm (GrEA) [3] and the constrained decomposition approach with grids for MOEA (CDG-MOEA) [4]. The two grid-based algorithms evenly partition each dimension (i.e., objective), which is beneficial to maintain diversity. However, with the increase in the number of objectives, the number of grids will increase exponentially, resulting in the curse of dimensionality.

In this paper, an MOEA based on space partitioning (MOEA-SP) is proposed to address the above issues. In MOEA-SP, subspaces, partitioned by a k-dimensional tree (kd-tree), are sorted according to a bi-indicator criterion defined in this paper. The two indicators refer to dominance degree and the niche count that measure convergence and diversity, respectively. The introduction of historical archive pushes the population toward the true PF and distributed evenly.

The rest of this paper is organized as follows. Section II describes the related work of MOEAs based on decomposition and grid decomposition. Section III gives the details about MOEA-SP. Section IV presents the experimental studies. Finally, Section V concludes this paper.

2 Related Work

This section discusses related work regarding the improvement of decomposition-based and grid-based MOEAs.

The weight vector generation method in MOEA/D makes the population size inflexible, and the distribution of the generated weight vector is not uniform. In order to address these issues, researchers have proposed some improved methods. For example, Fang *et al.* [5] proposed the combination of transformation and uniform design (UD) method. Deb *et al.* [6] proposed the two-layer weight vector generation method, including the boundary and inside layer, where the weight vectors of the inside layer are shrunk by a coordinate transformation, finally, the weight vectors of the boundary and inside layer are merged into a set of weight vectors.

The three decomposition methods of MOEA/D are sensitive to the shapes of PF, and it is difficult to solve the PF with the shape of nonconvex or extremely convex. For Penalty-based Boundary Intersection (PBI), it is also difficult to set the penalty parameters. The improved methods include inverted PBI (IPBI) [7], new penalty scheme [8], and angle penalized distance (APD) decomposition method [9]. Although these methods improve the performance of the algorithms for some MOPs, they are still sensitive to the shape of PF and difficult to set the penalty parameters. Besides, to overcome the shortcomings of traditional decomposition methods, Liu *et al.* [10] proposed a new alternative decomposition method, i.e., using a set of reference vectors to divide the objective space into multiple subspaces and assign a subpopulation to evolve in each subspace. This method does not need traditional decomposition methods.

The neighborhood could have a significant impact on generating offspring and environment selection, so the improper definition of the neighborhood relationship may mislead algorithms. To define the proper neighborhood, Zhao *et al.* [11] proposed to dynamically adjust the neighborhood structure according to the distribution of the current population. To address the issue of the loss of diversity due to the large update area in the selection of offspring, Li *et al.* [12] introduced the concept of the maximum number of a new solution to replace the old solutions. Zhang *et al.* [13] proposed a greedy-based replacement strategy, which calculates the improvement of the new solution for each subproblem, and then replaces the solutions of the two subproblems with the new solution with the best improvement.

Although there are many improved methods, they can not fundamentally overcome the above limitations. In order to overcome the limitations of the sensitivity for the shapes of PF and loss of diversity in decomposition-based method, Cai *et al.* [4] proposed a constrained decomposition approach with grids for MOEA (CDG-MOEA), which uniformly divides the objective space into $M \times K^{M-1}$ (where K is the grid decomposition parameter, and M is the number of objectives) subproblems, and choose offspring according to a decomposition-based ranking and lexicographic sorting method. CDG-MOEA has great advantages to maintain diversity. However, with the increasing number of objectives, the number of subproblems will increase exponentially, resulting in the curse of dimensionality.

3 MOEA Based on Space Partitioning

This section introduces the design process of the proposed algorithm. The major idea of MOEA-SP is to partition the objective space into a set of subspaces, then select offspring according to the rank of subspaces after sorting.

3.1 The Framework of MOEA-SP

Algorithm 1 gives the framework of MOEA-SP in detail. MOEA-SP starts with initialization, then repeat generate offspring, partition objective space, and select offspring (i.e., environmental selection) until the termination conditions are

satisfied. The environmental selection mainly includes nondominated sorting of the subspaces, subspace-orient selection, and Max-Min selection.

Algorithm 1 MOEA-SP

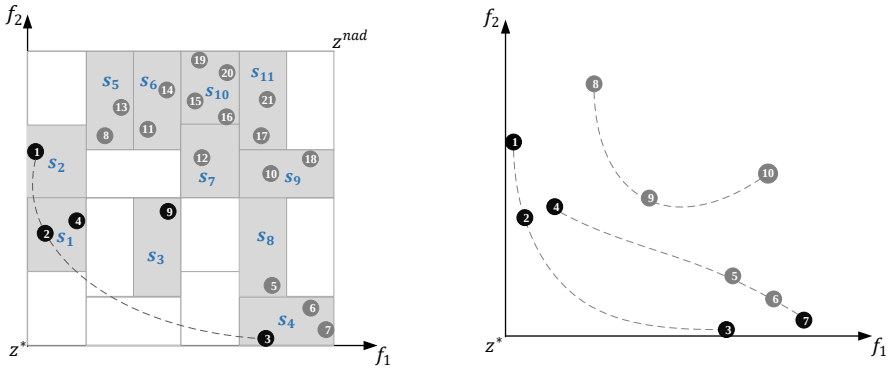
Input:
N: the population size of P;
K: number of subspace based on kd-tree partitioning method.
Output: A solution set P.
Step 1: Initialization
1.1 Initialize a population $P_0 = \{x^1, \ldots, x^N\}$ randomly;
1.2 $A = P_0$; // A is historic archive
1.3 Set $t = 0$.
Step 2: Reproduction
2.1 Update z^*, z^{nad} by P_t;
2.2 Build a kd-tree based on objective space formed by $[z^*, z^{nad}]$;
2.3 Obtain neighborhood based on kd-tree: $N(x), x \in P_t$;
2.4 Generate an empty set $Q_t = \emptyset$;
for all $x \in P_t$ **do**
2.5 Fill mating pools of x

$$M(x) = \begin{cases} N(x) & rand < \delta \ \ and \ \ |N(x)| > 3, \\ P_t & otherwise. \end{cases}$$

2.6 Select three solutions x^1, x^2 and x^3 randomly from $M(x)$, then generate offspring y by DE operator [12], and put y to Q_t.
end for
2.7 $A = A \cup Q_t$.
Step 3: Environmental selection
3.1 $P_{t+1} = \emptyset$.
3.2 Find nondominated solutions A^r, and $A^d = A \backslash A^r$;
3.3
if $|A^r| < N$ **then**
$P_{t+1} = P_t \cup A^r$;
Update z^*, z^{nad} by A;
Partition the objective space formed by $[z^*, z^{nad}]$ with a kd-tree;
R = SSBB (kd-tree); // Algorithm 2
P_{t+1} = DSOS (R, A^d, P_{t+1}); // Algorithm 4
Update A, delete some solutions from each subspace;
else
Update z^*, z^{nad} by A^r;
Partition the objective space formed by $[z^*, z^{nad}]$ with a kd-tree;
R = SSBB (kd-tree);
P_{t+1} = NSOMMS (R, A^r, P_{t+1}); // Algorithm 5
Update A, this is $A = A^r$;
end if
3.5 $t = t + 1$.
Step 4: Termination
If the stopping criterion is satisfied, terminate the algorithm. Otherwise, go to Step 2.

3.2 Subspace-oriented Domination and Sorting

In MOEA-SP, the multiple subspaces are obtained after the objective space is partitioned by kd-tree method. The set of subspaces is denoted as $R = \{s_1,\ s_2,\ \dots\ ,\ s_n\}$, where each subspace may contain a number of individuals. As shown in Fig.1a, the objective space is partitioned into twenty subspaces, eleven of them, which contain individuals, are marked with $s_1, ..., s_{11}$.



(a) Selection in MOEA-SP with $A^r < N$ (b) Selection in NSGA-II

Fig. 1. (a) Selection in MOEA-SP with $A^r < N$. (b) Selected in NSGAII. Solid circles represent individuals, where deeper-color circles represent individuals selected. Individuals connected by dotted lines are the same level of individuals after nondominated sorting. Note that in (a), there are individuals distributed in the gray subspaces (i.e, $s_1, ..., s_{11}$), and the total number of individuals in the historical archive is twenty-one.

This paper defines the concept of subspace-oriented: neighborhood $N(x)$, niche count (nc), dominance ratio (dr), dominance matrix (D) and dominance degree (dd).

The neighborhood $N(x)$ of individual x is defined as the set of individuals in the neighborhood of the subspace to which it belongs. It is worth noting that a subspace itself belongs to its own neighborhood. As shown in Fig.1a, for x_1 of belonging to s_2, the neighborhood of subspace s_2 are subspaces $s_1,\ s_2,\ s_5$, which contains five individuals: $x_1,\ x_2,\ x_4,\ x_8,\ x_{13}$, i.e., $N(x_1) = \{x_1,\ x_2,\ x_4,\ x_8,\ x_{13}\}$.

The niche count (nc) of the subspace is defined as the sum of all individuals in its adjacent subspaces in historical archive A. The niche count of all subspaces is recorded as $NC = \{nc(s_1), ..., nc(s_n)\}$. As shown in lines 9 to 12 of Algorithm 2. For example, the neighborhood of subspace s_2 includes subspaces s_1, s_2, and s_5, and they consist of five individuals with a fold of $x_1,\ x_2,\ x_4,\ x_8,$ and x_{13}, so $nc(s_2) = 5$. In this example, the niche count of all subspaces can be obtained, i.e., $NC = \{3,\ 5,\ 2,\ 4,\ 5,\ 9,\ 11,\ 7,\ 6,\ 9,\ 9\}$.

The dominance ratio (dr) is defined as

$$dr(s_1, s_2) = \frac{NO(s_1 \prec s_2) - NO(s_2 \prec s_1)}{M}. \tag{1}$$

where $NO(s_1 \prec s_2)$ denotes the number of objectives subspace s_1 dominates s_2, and M is the number of objectives. $dr(s_1, s_2)$ denotes the ratio of the difference

between the number of objectives s_1 dominates s_2 and the number of objectives s_2 dominates s_1 to M, obviously, $dr(s_1, s_2) = -dr(s_2, s_1)$. For example, subspaces s_1 and s_3 in the bi-objective MOP shown in Fig.1a, s_1 dominates s_3 on the second objective, and there is no comparability on the first one, so the difference in objectives number of s_1 dominance s_3 are 1, so $dr(s_1, s_3) = 1/M = 1/2$. If subspaces s_1 and s_2 do not dominate each other, then $dr(s_1, s_2) = 0$, such as subspaces s_1 and s_4 are nondominated; if subspace s_1 completely dominates s_2, then $dr(s_1, s_2) = 1$, for example, subspace s_1 completely dominates s_7.

The dominance matrix of the subspaces is defined as $D_{n\times n}$, where $D[i][j] = dr(s_i, s_j)$. In this example, as shown in Algorithm 2, the dominance matrix D is calculated as

$$D = \begin{pmatrix}
0 & \frac{1}{2} & \frac{1}{2} & 0 & 1 & 1 & 1 & \frac{1}{2} & 1 & 1 & 1 \\
-\frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & 0 & \frac{1}{2} & 1 & \frac{1}{2} \\
-\frac{1}{2} & 0 & 0 & 0 & 0 & \frac{1}{2} & 1 & \frac{1}{2} & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & \frac{1}{2} & \\
-1 & -\frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\
-1 & -\frac{1}{2} & -\frac{1}{2} & 0 & -\frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\
-1 & -\frac{1}{2} & -1 & 0 & -\frac{1}{2} & -\frac{1}{2} & 0 & 0 & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\
-\frac{1}{2} & 0 & -\frac{1}{2} & -\frac{1}{2} & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\
-1 & -\frac{1}{2} & -1 & -\frac{1}{2} & 0 & 0 & -\frac{1}{2} & -\frac{1}{2} & 0 & 0 & \frac{1}{2} \\
-1 & -1 & -1 & 0 & -\frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} \\
-1 & -\frac{1}{2} & -1 & -\frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} & 0
\end{pmatrix}.$$

Algorithm 2 Subspaces Sorting Based on Bi-indicator (SSBB)

Input: A kd-tree (K subspaces).
Output: New sorting of subspaces R.

$R = \{s \mid \exists$ solution $x \in s$th subspace$\}$, $n = |R|$;
Dominance matrix: $D_{n\times n}$; Dominance degree: $DD = \{dd(s_1), \ldots, dd(s_n)\}$;
Neighbor of sth subspace: $NR(s)$; Niche count: $NC = \{nc(s_1), \ldots, nc(s_n)\}$;
/* Calculate dominance degree DD */
Calculate $D_{n\times n}$ according to dominance ratio in equation (1)
for $s = 1$ to n **do**
  $GP(l) = \{l|D[s][l] > 0\}$,$GN(l) = \{l|D[s][l] < 0\}$;
  $val_s = \sum_{l\in GP(l)}^{n} D[s][l]$;
  $val\_ed_s = \sum_{l\in GN(l)}^{n} D[s][l]$;
  $dd(s) = val_s/val\_ed_s$;
end for
/* Calculate niche count NC */
for $s = 1$ to n **do**
  $NR(s) = \{s_1, s_2, ...\}$;
  $nc(s) = \sum_{i\in NR(s)} |subspace(s)_{ind}|$;
end for
$Rank(s) = Nondominated\text{-}sorting(DD, NC)$; // maximum dd and minimum nc
$R = sort(R, Rank(s))$. // ascending sort R according to rank

The dominance degree (dd) of subspace denotes the degree of convergence, calculated according to the dominance matrix D. The dominance degree of all

subspaces is recorded as $DD = \{dd(s_1), dd(s_2), ..., dd(s_n)\}$. As shown in lines 3 to 8 of Algorithm 2, according to the domination matrix, the dominating value val_s and the dominated value val_ed_s of each subspace can be obtained, i.e., $val_s = \sum_{l \in GP(l)}^{n} D[s][l]$ $(GP(l) = \{l \mid D[s][l] \geq 0\})$, $val\_ed_s = \sum_{l \in GN(l)}^{n} |D[s][l]|$ $(GN(l) = \{l \mid D[s][l] \leq 0\})$, and then the dominance degree of the subspace is calculated by $dd(s) = val_s/val\_ed_s$. Note that when val or val_ed is 0, dd is assigned to $+1e5$ or $-1e5$, respectively. In this example, $val_2 = D[2][5] + D[2][6] + D[2][7] + D[2][8] + D[2][9] + D[2][10] = 7/2$, $val\_ed_2 = |D[2][1]| = 1/2$, then $dd(s_2) = val_2/val\_ed_2 = 7$. Finally, DD can be obtained, i.e., $DD = \{+1e5,\ 7,\ 10,\ +1e5,\ 4/3,\ 3/5,\ 3/7,\ 2/3,\ 1/8,\ 1/9,\ -1e5\}$.

In summary, the dominance degree dd and niche count nc can measure the convergence and diversity, respectively and push the population to these two directions. This paper uses the two indicators as two objectives, i.e.,, maximizing dd, minimizing nc, and then performs nondominated sorting by these two indicators. As shown in lines 13 to 14 of Algorithm 2, a new sorting subspaces set R is obtained according to the rank value of nondominated sorting. In this example, according to the DD and NC values calculated above, the eleven subspaces can be divided into eight ranks, $\{s_1, s_3\}, \{s_4\}, \{s_2\}, \{s_5\}, \{s_8, s_9\}, \{s_6\}, \{s_7, s_{10}\}, \{s_{11}\}$ by nondominated sorting, i.e., $R = \{s_1, s_3, s_4, s_2, s_5, s_8, s_9, s_6,\ \ s_7, s_{10}, s_{11}\}$.

3.3 Environmental Selection

This paper uses the historical archive to find the nondominated individuals for the selection of offspring, where the set of nondominated individuals is recorded as A^r, and the set of other individuals is A^d.

Algorithm 3 Subspace-oriented Selection (SOS)

Input:
R: Sorted subspaces;
$A^{'}$: A set of candidate solutions;
P: A set of solutions;
num_max: The maximum selected from each subspace;
Output:
A set of solutions P;
The index of subspace s.
1: Let $s = 1$;
// N denotes the population size
2: **while** $|P| + min\{|R(s)_{inds}|,\ num\_max\} \leq N$ **do**
3: **if** $|R(s)_{ind}| \leq num\_max$ **then**
4: $P = P \cup R(s)_{inds}$;
5: **else**
6: *Nondominated selection* $(R(s)_{inds})$;
7: $P = P \cup R(s)_{inds}[1 : (num\_max)]$; // select num_max solutions from $R(s)_{inds}$
8: **end if**
9: $s = s + 1$;
10: **end while**

The environmental selection is divided into two cases according to the size of $|A^r|$, as shown in step 3.3 of Algorithm 1. IF $|A^r| < N$, then besides A^r, $N - |A^r|$ individuals need to be selected from A^d for the next-generation population; Otherwise, N individuals of the next-generation population need to be selected from A^r.

In either case of the above, the next-generation population is selected according to the sorted subspaces. Algorithm 3 shows subspace-oriented selection in detail, where num_max is the maximum number of individuals selected from each subspace. Firstly, we select the individuals from the first subspace in R, and proceed in sequence but not more than N (population size). To maintain diversity, this algorithm sets the maximum number of individuals (num_max) selected from each subspace. If the number of individuals in a subspace is not greater than num_max, then all the individuals in this subspace are selected; Otherwise, num_max individuals are selected according to the NSGAII [1].

Selection From Dominated Subspaces. In the first case, the number of nondominated individuals is smaller than N ($|A^r| < N$). $N - |A^r|$ individuals need to be selected from A^d for next-population, as mentioned earlier. The subspaces set R sorted was obtained by Algorithm 2. The next step is to select individuals from A^d distributed in dominated subspaces, the detailed procedure is shown in Algorithm 4. Note that if individuals in each subspace are not evenly distributed, the number of selected individuals will be less than N. At this time, we need to select the remaining individuals from the unselected A^d according to the NSGA-II [1], as shown in lines 6 to 7 of Algorithm 4.

As shown as Fig.1a, the result of environment selection for MOEA-SP is $P = \{x_1,\ x_2,\ x_3,\ x_4,\ x_9\}$, compared with the result $P = \{x_1,\ x_2,\ x_3,\ x_4,\ x_7\}$ of NSGA-II in Fig.1b. From the results of the two selection, the diversity of MOEA-SP is better than that of NSGA-II during evolution.

Algorithm 4 Dominated Subspaces-oriented Selection (DSOS)

Input:
R: Sorted subspaces;
A^d: A set of candidate solutions;
P: A set of solutions.
Output: A set of solutions P.
1: Let $s = 1$, $num\_max = 5$;
2: $(P,\ s) = \text{SOS}\ (R,\ A^d,\ P,\ num\_max)$; // Algorithm 3
3: **if** $(s \leq |R|)\&\&(|P| < N)$ **then**
4: $Nondominated\ selection\ (R(s)_{inds})$;
5: $P = P \cup R(s)_{inds}[1 : (N - |P|)]$;
6: **else if** $s > |R|\&\&|P| < N$ **then**
7: Select $(N - |P|)$ solutions from the unselected solution from A^d;
8: **end if**

Selection From Nondominated Subspaces. In the second case, the number of nondominated individuals is larger than N ($A^r \geq N$). N individuals of next-generation population need to be selected from A^r. As shown in Fig.2, $|A^r| = 7 > N = 5$. In this case, the population normally enters into the late stage of the evolution. This means diversity should be paid more attention than convergence.

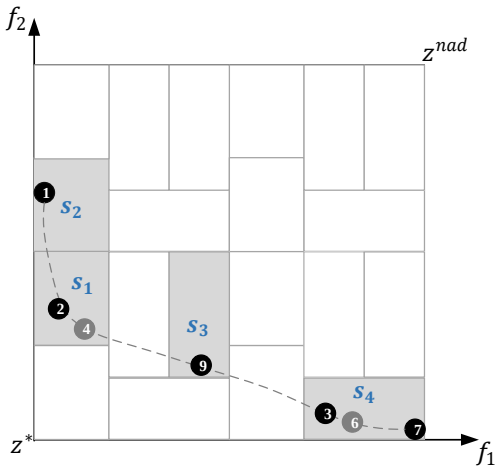


Fig. 2. Nondominated Subspaces based Max-Min Selection

Algorithm 5 Nondominated Subspaces-oriented Max-Min Selection (NSOMMS)

Input:
R: Sorted subspaces;
A^r: A set of candidate solutions;
P: A set of solutions;
Output: A set of solutions P.

Let $num\_max = \left\lfloor \frac{N}{|R|} \right\rfloor$;
$P = \text{SOS}\ (R,\ A^d,\ P,\ num\_max)$; // Algorithm 3
$A^r = A^r \setminus P$
/* Select lacked individuals by $Max\text{-}Min$ method */
while $|P| < N$ **do**
  Let $Dist_{1\times|A^r|}$, $i = 1$;
  for all $x \in A^r$ **do**
    $Dist[i] = \min_{y\in P}\ distance\_Minkowski(x, y)$;
    $i = i + 1$;
  end for
  $q = arg\ \max_{x\in A^r} Dist[x_{index}]$;
  $P = P \cup \{q\}$;
  $A^r = A^r \setminus \{q\}$;
end while

Different from Algorithm 4, in Algorithm 5, num_max is the average number of individuals in each subspace, which is set to $num\_max = \left\lfloor \frac{N}{|R|} \right\rfloor$. In this way, the value of num_max will be adaptively adjusted according to the distribution of the current population. The setting of num_max ensures that individuals will be selected from each nondominated subspace, which is conducive to maintaining the diversity of the population.

Max-Min selection is described with lines 4 to 13 of Algorithm 5. The major idea is to select an individual each time that its minimum distance to the set of selected individuals is the largest. In other words, an individual, who is as far away as possible from the selected individual, is expected to select. Firstly, the minimum Minkowski distance between each individual in A^r and individual in P is calculated and stored in $Dist$. Then, the individual in A^r corresponding to the maximum value of $Dist$ is found and selected into P, which proceeds in turn until the number of individuals in P reaches N. The method is conducive to maintaining diversity greatly. Here the Minkowski distance is calculated as

$$distance\_Minkowski\ (x, y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{\frac{1}{p}} \qquad (0 < p < 1). \tag{2}$$

where n is the dimension of x or y, p is a parameter. The reason why the Minkowski distance $(0 < p < 1)$ is used is that it can enlarge the difference and facilitate the comparison of distances.

For example in Fig.2, $num\_max = \left\lfloor \frac{5}{4} \right\rfloor = 1$, and individuals $x_1,\ x_2,\ x_9,\ x_3$ are selected from $s_1,\ s_2,\ s_3,\ s_4$, respectively, i.e., $P = \{x_1,\ x_2,\ x_9,\ x_3\}$. It is necessary to select an individual from $\{x_4,\ x_6,\ x_7\}$ according to the following Max-Min method because of $5-|P| = 1$. Obviously, individual x_7 in $\{x_4,\ x_6,\ x_7\}$ is the farthest away from all the selected individuals. So individual x_7 is selected, i.e., $P = \{x_1,\ x_2,\ x_9,\ x_3,\ x_7\}$.

3.4 Historical Archive Update

The reason for introducing the historical archive is that it can guide the direction of population evolution and facilitate the convergence and diversity of the population. The historical archive participates in the calculation of the aforementioned two indicators and environmental selection. In the early stage of evolution, due to the pressure and information of historical individuals, it promotes the convergence of population and accelerates the evolution speed and efficiency. In the late stage, it can increase diversity to select offspring from the set of all nondominated historical individuals according to the Max-Min method. However, if the number of historical individuals is too large, computing resources will be overtaken, so we need to delete some individuals from the historical archive.

The update of the historical archive in this paper is divided into two cases according to the size of nondominated individuals. In the first case, the number of nondominated individuals is smaller than the size of the population, which needs historical individuals to guide algorithm search. Therefore, the historical

archive reserves some representative individuals in each sampled subspace. In the second case, when the number of nondominated individuals is larger than the size of the population, the historical archive only preserves nondominated individuals.

4 Experimental Studies

To verify the validity of the proposed algorithm MOEA-SP, this paper makes some comparative experiments on a set of benchmarks.

4.1 Benchmark Functions and Performance Metric

GLT1-GLT6 [14] benchmark problems are used for testing the performance of algorithms. The shape of these functions has a variety of forms, including convex, nonconvex, extremely convex, disconnected, nonuniformly distributed. The dimensions of the decision variables of GLT1-GLT6 are set to 10.

In this paper, the Inverted Generational Distance (IGD) is used as a performance metric to measure the quality of a solution set P, which represents the average distance from a set of reference points P^* on true PF to the solved population P. The IGD metric is defined as

$$IGD(P^*, P) = \frac{\sum_{v \in P^*} dist(v, P)}{|P^*|}. \tag{3}$$

where $dist(v, P)$ is the minimum Euclidean distance from the solution v in P^* to solution in P. The IGD metric can measure the convergence and diversity of a set of solutions P, and the smaller the value of the IGD is, the better the algorithm performs.

4.2 Peer Algorithms and Parameter Settings

The compared algorithms with MOEA-SP are CDG-MOEA [4], NSGA-II [1] and MOEA/D [2]. These MOEAs belong to grid-based, dominance-based and decomposition-based methods, respectively. The population size is set to 200. Here the population size, in MOEA/D, is set to 210 (the closest integer to 200) for three objective problems because of the same as the number of weight vectors.

The number of evaluations of each test function is: 300,000; $\delta = 0.9$; In the DE operator: $CF = 1$, $F = 0.5$, $\eta = 20$, $p_m = 1/D$, where D is the dimensions of the decision variables. Each of the above four MOEAs runs 30 times independently on each test problem.

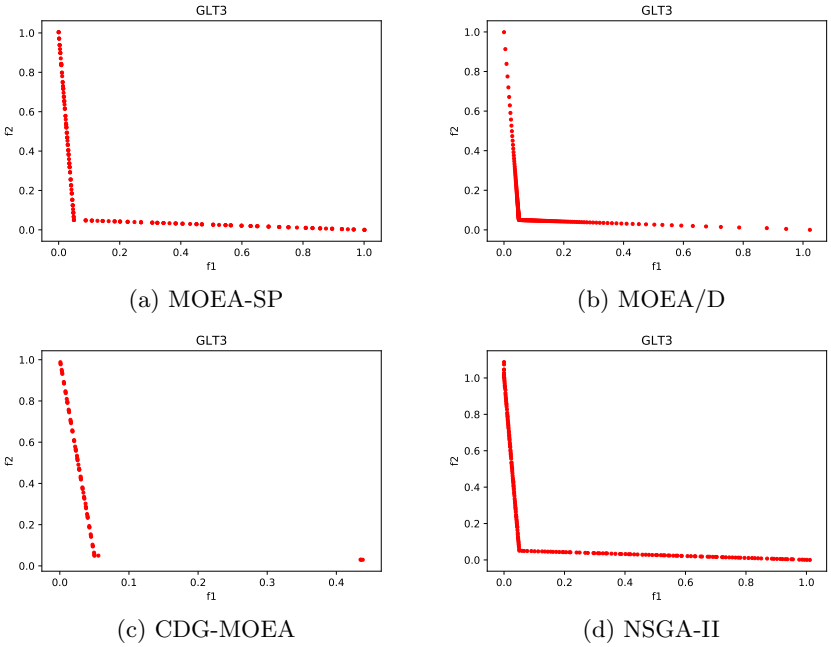
4.3 Experimental Results

Table 1 gives the mean and standard deviation of the IGD-metric values of the four algorithms (i.e., MOEA-SP, CDG-MOEA, NSGA-II, and MOEA/D) on GLT1-GLT6 test problems in 30 independent runs, where the IGD-metric value

with the best mean for each test problem is highlighted in boldface. MOEA-SP performs the best on GLT5 and GLT6 test problems, compared with the other three MOEAs. For GLT1-GLT4 test problems, the other three MOEAs have their own strong points. NSGA-II performs the best on GLT2 and GLT3. CDG-MOEA and MOEA/D perform the best on GLT1 and GLT4, respectively.

Table 1. The IGD-metric values comparisons of the MOEA-SP with the other three MOEAs on GLT test problems in terms of the mean and standard deviation values, where the IGD-metric value with the best mean for each test problem is highlighted in boldface.

		MOEA-SP	CDG-MOEA	NSGA-II	MOEA/D
GLT1	mean	4.30E-03	**1.55E-03**	1.76E-03	1.78E-03
	std	7.25E-04	**6.45E-04**	1.71E-04	5.58E-06
GLT2	mean	3.64E-02	1.03E-01	**1.66E-02**	1.45E-01
	std	1.54E-03	6.63E-02	**4.41E-04**	1.83E-02
GLT3	mean	5.96E-03	1.03E-01	**5.48E-03**	9.18E-03
	std	5.16E-04	8.45E-02	**6.22E-03**	1.42E-04
GLT4	mean	1.91E-02	6.59E-03	9.79E-03	**4.78E-03**
	std	4.06E-02	3.56E-03	3.37E-02	**8.17E-06**
GLT5	mean	**2.91E-02**	2.52E-01	4.18E-02	7.86E-02
	std	**5.05E-04**	1.18E-01	1.87E-03	3.98E-04
GLT6	mean	**2.93E-02**	1.90E-01	2.97E-02	4.56E-02
	std	**5.32E-03**	4.88E-02	1.65E-03	5.21E-04

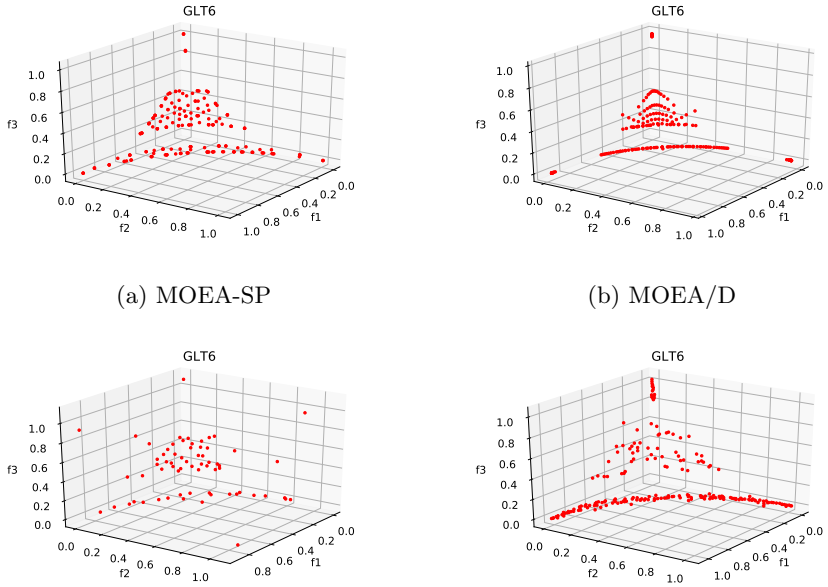


(a) MOEA-SP (b) MOEA/D

(c) CDG-MOEA (d) NSGA-II

Fig. 3. The final nondominated solution set obtained by the four algorithms on GLT3

The reason for the best performance of CDG-MOEA on GLT1 is that the grid partitioning is very favorable for GLT1 with the linear shape of PF, compared with the kd-tree partitioning in the MOEA-SP. Although NSGA-II performs the best on GLT3 with the extremely concave PF, MOEA-SP does not perform poorly. As shown in Fig.3, we give the distribution of the nondominated solutions sets obtained by the four algorithms on GLT3. In terms of the uniformity, the distribution of the nondominated solutions set obtained by MOEA-SP is better than MOEA/D. The performance of MOEA-SP on GLT5 and GLT6 is superior to the other three MOEAs. The reason is that, in MOEA-SP, the subspace sorting and the subspace-oriented selection increase selection pressure and are conducive to solving GLT5 and GLT6 with three objectives. As shown in Fig.4, we give the distribution of the nondominated solutions sets obtained by the four algorithms on GLT6. The distribution of nondominated solutions set obtained by MOEA-SP on GLT6 is superior to the other three algorithms in terms of both the convergence and diversity.



(a) MOEA-SP (b) MOEA/D

(c) CDG-MOEA (d) NSGA-II

Fig. 4. The final nondominated solution set obtained by the four algorithms on GLT6

5 Conclusion

This paper proposes a novel MOEA based on space partitioning (MOEA-SP). The proposed MOEA-SP transforms a MOP into a bi-objectives optimization problem to sort the subspaces, then selects offspring by the subspace-oriented se-

lection, which simplifies the complexity of the problem. The kd-tree space partitioning method overcomes the limitation of shape sensitivity to PF and the curse of dimensionality. The subspaces sorting and the Max-Min selection are used for pushing the population convergence and maintaining the diversity, respectively, which has great potential to solve MaOPs. MOEA-SP is compared with three MOEAs on GLT test suite. The experimental results show that MOEA-SP outperforms the compared algorithms in some benchmarks.

Although MOEA-SP overcomes some of the above issues, it still faces great challenges. If the number of partitioning subspaces and the value of num_max are not set properly, the performance of the algorithm will be affected to some extent. In addition, the definition of the bi-indicator criterion needs to fine-tune to the more accurate description of the convergence and the diversity. Adaptively adjusting the num_max and furtherly modifying the definition of the bi-indicator criterion what needs to be done in the future.

Acknowledgments. This work was supported in part by the National Natural Science Foundation of China under Grant No. 61673355, in part by the Fundamental Research Funds for the Central Universities, China University of Geosciences (Wuhan)under Grant(CUG170603,CUGGC02).

References

1. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. 6(2), 182–197 (2002).
2. Zhang, Q., Hui L.: MOEA/D: A multiobjective evolutionary algorithm based on decomposition. IEEE Trans. Evol. Comput. 11(6), 712–731 (2007)
3. Yang, S., Li, M., Liu, X., Zheng, J.: A grid based evolutionary algorithm for many objective optimization. IEEE Trans. Evol. Comput. 17(5), 721–736 (2013)
4. Cai, X., Mei, Z., Fan, Z., Zhang, Q.: A constrained decomposition approach with grids for evolutionary multiobjective optimization. IEEE Trans. Evol. Comput. 22(4), 564–577 (2017)
5. Fang, K., Yang, Z.: On uniform design of experiments with restricted mixtures and generation of uniform distribution on some domains. Stat. Probab. Lett. 46(2), 113–120 (2000)
6. Deb, K., Jain, H.: An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints. IEEE Trans. Evol. Comput. 18(4), 577–601 (2014)
7. Sato, Hiroyuki.: Analysis of inverted PBI and comparison with other scalarizing functions in decomposition based MOEAs. J. Heuristics. 21(6), 819–849 (2015)
8. Yang, S., Jiang, S., Yong, J.: Improving the multiobjective evolutionary algorithm based on decomposition with new penalty schemes. Soft. Comput. 21(16), 1–15 (2016)
9. Cheng, R., Jin, Y., Olhofer, M., Sendhoff, B.: A reference vector guided evolutionary algorithm for many-objective optimization. IEEE Trans. Evol. Comput. 20(5), 773–791 (2016)
10. Liu, H., Gu, F., Zhang, Q.: Decomposition of a multiobjective optimization problem into a number of simple multiobjective subproblems. IEEE Trans. Evol. Comput. 18(3) 450–455 (2014)

11. Zhao, S., Suganthan, P., Zhang, Q.: Decomposition-based multiobjective evolutionary algorithm with an ensemble of neighborhood sizes. IEEE Trans. Evol. Comput. 16(3), 442–446 (2012)
12. Li, H., Zhang, Q.: Multiobjective optimization problems with complicated pareto sets, MOEA/D and NSGA-II. IEEE Trans. Evol. Comput. 13(2), 284–302 (2009).
13. Zhang, H., Zhang, X., Gao, X., Song, S.: Self-organizing multiobjective optimization based on decomposition with neighborhood ensemble. Neurocomputing. 173, 1868–1884 (2016)
14. Gu, F., Liu, H., Tan, K.: A multiobjective evolutionary algorithm using dynamic weight design method. Int. J. Innov. Comput. I. 8(5), 3677–3688 (2012)